Package: boundedHill (via r-universe)

October 21, 2026

Type Package

Title Bayesian Logistic Regression with Lower and Upper Asymptotes

Version 0.1.0

Description Fits Bayesian logistic regression models in which the success probability rises or falls between a lower and an upper asymptote instead of between 0 and 1, for binomial outcomes including binary data. The success probability equals the lower asymptote plus the difference between the two asymptotes multiplied by the logistic function of a linear predictor. The lower and upper asymptotes can be either estimated or fixed. With a single predictor such as log dose, this is the four-parameter logistic (Hill) model. Posterior computation is a full Gibbs sampler that requires no tuning.

License GPL (>= 3)

Encoding UTF-8

Language en-US

Depends R (>= 4.1.0)

Imports BayesLogit, graphics, grDevices, posterior, stats

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

Config/roxygen2/version 8.1.0

NeedsCompilation no

Author Minji Lee [aut, cre, cph], Chang Hyeong Lee [aut, cph], Ryung S. Kim [aut, cph]

Maintainer Minji Lee <leeminji@unist.ac.kr>

Repository https://cran.r-universe.dev

Date/Publication 2026-10-21 14:40:08 UTC

RemoteUrl https://github.com/cran/boundedHill

RemoteRef HEAD

RemoteSha 71e496d4cbc6be179b7cdfb1ecd3154483fc2380

Contents

bh_beta_moments 2
bh_control 3
bh_diagnose 3
bh_fit 4
bh_log_lik 6
bh_posterior_draws 7
bh_posterior_predict 7
bh_prior 8
bounded_probability 9
plot.bounded_reg 10
print.bounded_reg 11
rbounded_binom 13

Index **14**

`bh_beta_moments` *Beta shapes from a prior mean and standard deviation*

Description

Convert a prior mean and standard deviation for an asymptote into the two Beta shape parameters used by `bh_prior`. With mean m and standard deviation s, the shapes are $a = mk$ and $b = (1-m)k$ with $k = m(1-m)/s^2 - 1$.

Usage

```
bh_beta_moments(mean, sd)
```

Arguments

`mean`	Prior mean, strictly between 0 and 1.
`sd`	Prior standard deviation; must satisfy `sd^2 < mean * (1 - mean)`.

Value

A named numeric vector `c(shape1, shape2)`.

Examples

```
bh_beta_moments(0.10, 0.12)   # baseline response around 10 percent
bh_beta_moments(0.85, 0.12)   # plateau around 85 percent
bh_prior(delta0 = bh_beta_moments(0.2, 0.1),
         delta1 = bh_beta_moments(0.7, 0.1))
```

bh_control — *Control Gibbs sampling*

Description

Set the number of iterations, warmup, thinning, and chains used by `bh_fit`.

Usage

```
bh_control(iter = 4000L, warmup = 2000L, thin = 1L, chains = 4L)

## S3 method for class 'bh_control'
print(x, ...)
```

Arguments

`iter`	Total iterations per chain, including warmup.
`warmup`	Number of initial iterations discarded per chain.
`thin`	Positive thinning interval.
`chains`	Number of chains.
`x`	A `bh_control` object.
`...`	Further arguments; currently unused.

Value

An object of class `bh_control`: a list with integer elements `iter`, `warmup`, `thin` and `chains`. The `print` method returns its input invisibly.

Examples

```
bh_control(iter = 2000, warmup = 1000, chains = 2)
```

bh_diagnose — *Diagnose computation and data-range information*

Description

Check MCMC convergence and whether the data identify the bounded logistic curve. Diagnostics use the original, unmodified posterior chains. Endpoints fixed by the prior are excluded from convergence flags. Undefined diagnostics on other parameters are flagged rather than treated as evidence of convergence. Switching of the endpoint labels can give poor diagnostics for the raw parameters even when the fitted probabilities (see `predict.bounded_reg`) mix well.

Usage

```
bh_diagnose(object, rhat_threshold = 1.01, ess_threshold = 400)

## S3 method for class 'bh_diagnostic'
print(x, digits = 3L, ...)
```

Arguments

`object`	A `bounded_reg` fit.
`rhat_threshold`	R-hat warning threshold.
`ess_threshold`	Bulk and tail ESS warning threshold.
`x`	A `bh_diagnostic` object.
`digits`	Number of displayed digits.
`...`	Further arguments; currently unused.

Value

An object of class `bh_diagnostic`: a list with the MCMC summary table (`mcmc`), the names of fixed and flagged parameters, the design-matrix condition number, posterior quantiles of the logistic gate span and of the endpoint gap `delta1 - delta0`, the posterior probability that `delta0 > delta1`, the number of endpoint-order switches per chain, and a character vector of warning messages.

Examples

```
set.seed(1)
x <- seq(-3, 3, length.out = 60)
y <- rbounded_binom(-0.5 + 1.2 * x, delta0 = 0.1, delta1 = 0.8)
fit <- bh_fit(y ~ x, data = data.frame(x = x, y = y),
              control = bh_control(iter = 400, warmup = 200, chains = 2),
              seed = 1)
bh_diagnose(fit)
```

`bh_fit` *Fit Bayesian bounded logistic regression*

Description

Fit a bounded logistic regression model to Bernoulli or grouped binomial outcomes using a fully Gibbs posterior sampler. The right-hand side of `formula` is processed by `model.matrix`, so continuous predictors, factors, interactions, and standard transformations are supported.

Usage

```
bh_fit(
  formula,
  data,
  prior = bh_prior(),
  control = bh_control(),
  seed = NULL,
  subset,
  na.action = stats::na.omit,
  contrasts = NULL
)
```

Arguments

formula	Model formula. Use y ~ x1 + x2 for Bernoulli data or cbind(successes, failures) ~ x1 + x2 for grouped binomial data.
data	Data frame containing variables in formula.
prior	Prior specification from bh_prior.
control	Sampling controls from bh_control.
seed	Optional integer seed. Chain-specific seeds are generated from it, so the same seed reproduces the same draws. Fitting changes the state of the random number generator in the calling session, also when seed = NULL.
subset	Optional row subset passed to model.frame.
na.action	Missing-value action.
contrasts	Optional contrasts passed to model.matrix.

Value

An object of class bounded_reg: a list containing the call, formula, model matrix, response counts, prior, control settings, chain seeds, and the posterior draws (a **posterior** draws_array in element draws). Use the methods in bounded_reg-methods, plot.bounded_reg and bh_diagnose to work with it.

References

Polson, N. G., Scott, J. G., and Windle, J. (2013). Bayesian inference for logistic models using Polya-Gamma latent variables. *Journal of the American Statistical Association*, 108(504), 1339–1349. doi:10.1080/01621459.2013.829001

Lee, M., Lee, C. H., and Kim, R. S. A novel Bayesian phase 1/2 design incorporating the Hill equation. Manuscript in preparation.

Lee, M., Lee, C. H., and Kim, R. S. boundedHill: Bayesian logistic regression with lower and upper asymptotes in R. Manuscript in preparation.

Examples

```
set.seed(1)
x <- seq(-3, 3, length.out = 60)
y <- rbounded_binom(-0.5 + 1.2 * x, delta0 = 0.1, delta1 = 0.8)
fit <- bh_fit(y ~ x, data = data.frame(x = x, y = y),
              control = bh_control(iter = 400, warmup = 200, chains = 2),
              seed = 1)
summary(fit)

# Grouped binomial data
grouped <- data.frame(dose = c(50, 100, 150, 200, 250),
                      resp = c(2, 11, 14, 15, 14), n = c(15, 20, 21, 22, 20))
fit2 <- bh_fit(cbind(resp, n - resp) ~ log(dose / 150), data = grouped,
               control = bh_control(iter = 400, warmup = 200, chains = 2),
               seed = 2)
coef(fit2)
```

---

bh_log_lik *Pointwise log-likelihood draws*

Description

Compute the binomial log likelihood of each observation under each posterior draw, for example for use with the **loo** package.

Usage

```
bh_log_lik(object, newdata = NULL)
```

Arguments

object	A bounded_reg fit.
newdata	Optional new data containing both the response and the predictors. If NULL, the data used for fitting are used.

Value

A numeric matrix with posterior draws in rows and observations in columns.

Examples

```
set.seed(1)
x <- seq(-3, 3, length.out = 60)
y <- rbounded_binom(-0.5 + 1.2 * x, delta0 = 0.1, delta1 = 0.8)
fit <- bh_fit(y ~ x, data = data.frame(x = x, y = y),
              control = bh_control(iter = 400, warmup = 200, chains = 2),
              seed = 1)
ll <- bh_log_lik(fit)
dim(ll)
```

bh_posterior_draws *Extract posterior draws*

Description

Extract posterior draws

Usage

```
bh_posterior_draws(object, format = c("draws_array", "matrix", "data.frame"))
```

Arguments

object	A bounded_reg fit.
format	One of "draws_array" (a **posterior** draws_array), "matrix", or "data.frame".

Value

Posterior draws in the requested format, with the asymptotes delta0, delta1 and the regression coefficients beta_* as variables.

Examples

```
set.seed(1)
x <- seq(-3, 3, length.out = 60)
y <- rbounded_binom(-0.5 + 1.2 * x, delta0 = 0.1, delta1 = 0.8)
fit <- bh_fit(y ~ x, data = data.frame(x = x, y = y),
              control = bh_control(iter = 400, warmup = 200, chains = 2),
              seed = 1)
draws <- bh_posterior_draws(fit, format = "matrix")
dim(draws)
```

bh_posterior_predict *Draw from the posterior predictive distribution*

Description

Draw from the posterior predictive distribution

Usage

```
bh_posterior_predict(
  object,
  newdata = NULL,
  trials = NULL,
  ndraws = 100L,
  seed = NULL
)
```

Arguments

`object`	A `bounded_reg` fit.
`newdata`	Optional data frame of predictor values. If `NULL`, the data used for fitting are used.
`trials`	Trial counts for `newdata`; defaults to one. For the original data, the observed trial counts are used.
`ndraws`	Number of posterior predictive draws, or `NULL` to use all stored posterior draws.
`seed`	Optional random seed.

Value

An integer matrix with predictive draws in rows and observations in columns.

Examples

```
set.seed(1)
x <- seq(-3, 3, length.out = 60)
y <- rbounded_binom(-0.5 + 1.2 * x, delta0 = 0.1, delta1 = 0.8)
fit <- bh_fit(y ~ x, data = data.frame(x = x, y = y),
              control = bh_control(iter = 400, warmup = 200, chains = 2),
              seed = 1)
yrep <- bh_posterior_predict(fit, ndraws = 10, seed = 4)
dim(yrep)
```

`bh_prior` *Specify priors for bounded logistic regression*

Description

Construct independent Beta priors for the two endpoints and a multivariate normal prior for regression coefficients. No ordering constraint is imposed. A scalar fixes an endpoint; two positive values specify Beta shapes, which can be obtained from a prior mean and standard deviation with `bh_beta_moments`. The defaults put the baseline response (`delta0`) at mean 0.10 and the plateau (`delta1`) at mean 0.85, both with standard deviation 0.12, and give each regression coefficient an independent normal prior with mean 0 and standard deviation 2.

Usage

```
bh_prior(
  delta0 = bh_beta_moments(0.1, 0.12),
  delta1 = bh_beta_moments(0.85, 0.12),
  beta_mean = 0,
  beta_sd = 2,
  beta_cov = NULL
)

## S3 method for class 'bh_prior'
print(x, ...)
```

Arguments

`delta0`	Length-two Beta shape vector, or a scalar fixed eta-minus-infinity asymptote. Use `bh_beta_moments(mean, sd)` to give a mean and SD.
`delta1`	Length-two Beta shape vector, or a scalar fixed eta-plus-infinity asymptote. Use `bh_beta_moments(mean, sd)` to give a mean and SD.
`beta_mean`	Prior mean for regression coefficients. A scalar is recycled after the model matrix is known.
`beta_sd`	Prior standard deviation for independent regression coefficients. Ignored when `beta_cov` is supplied.
`beta_cov`	Optional prior covariance matrix. A scalar creates a diagonal covariance matrix with that variance.
`x`	A `bh_prior` object.
`...`	Further arguments; currently unused.

Details

`delta0` and `delta1` are the limits as the linear predictor tends to minus and plus infinity, respectively; they need not be the lower and upper probability bounds. Swapping the endpoints and negating `beta` leaves the likelihood unchanged, so symmetric priors produce a label-symmetric posterior. Asymmetric priors fix the labels.

Value

An object of class `bh_prior`: a list with elements `delta0`, `delta1`, `beta_mean`, `beta_sd` and `beta_cov`. The `print` method returns its input invisibly.

Examples

```
bh_prior()
# Asymptotes from mean and SD, wider N(0, 4^2) priors for the logistic part
bh_prior(delta0 = bh_beta_moments(0.15, 0.10),
         delta1 = bh_beta_moments(0.75, 0.10), beta_sd = 4)
# Standard logistic regression: both asymptotes fixed
bh_prior(delta0 = 0, delta1 = 1)
```

`bounded_probability` *Bounded logistic probabilities*

Description

Evaluate `delta0 + (delta1 - delta0) * plogis(eta)`. No ordering of the two endpoints is required.

Usage

```
bounded_probability(eta, delta0, delta1)
```

Arguments

`eta`	Finite numeric linear predictor.
`delta0`	Endpoint at eta tending to minus infinity, in [0, 1].
`delta1`	Endpoint at eta tending to plus infinity, in [0, 1].

Value

A numeric vector of probabilities. Inputs must be scalar or have a common length.

Examples

```
bounded_probability(c(-2, 0, 2), delta0 = 0.1, delta1 = 0.8)
```

`plot.bounded_reg` *Plot a bounded logistic regression fit*

Description

Draw MCMC trace plots, marginal posterior densities, or fitted probabilities with credible bands against the observed proportions.

Usage

```
## S3 method for class 'bounded_reg'
plot(x, type = c("fitted", "trace", "density"), variables = NULL, ...)
```

Arguments

`x`	A `bounded_reg` fit.
`type`	Plot type: `"fitted"` (fitted probabilities and observed proportions), `"trace"` (MCMC traces by chain), or `"density"` (marginal posterior densities).
`variables`	Optional character vector of parameter names for trace or density plots. Defaults to all parameters.
`...`	Additional graphical arguments passed to `plot`.

Value

The input object, invisibly. Called for its side effect of drawing a plot.

Examples

```
set.seed(1)
x <- seq(-3, 3, length.out = 60)
y <- rbounded_binom(-0.5 + 1.2 * x, delta0 = 0.1, delta1 = 0.8)
fit <- bh_fit(y ~ x, data = data.frame(x = x, y = y),
              control = bh_control(iter = 400, warmup = 200, chains = 2),
              seed = 1)
plot(fit)
plot(fit, type = "trace", variables = c("delta0", "delta1"))
```

---

print.bounded_reg *Methods for bounded logistic regression fits*

Description

Standard S3 methods for objects of class bounded_reg returned by bh_fit: printing, posterior summaries, point estimates, predictions, fitted values, and posterior predictive simulation.

Usage

```
## S3 method for class 'bounded_reg'
print(x, ...)

## S3 method for class 'bounded_reg'
summary(object, ...)

## S3 method for class 'summary.bounded_reg'
print(x, digits = 3L, ...)

## S3 method for class 'bounded_reg'
coef(object, statistic = c("mean", "median"), ...)

## S3 method for class 'bounded_reg'
predict(
  object,
  newdata = NULL,
  type = c("response", "linear", "draws"),
  level = 0.95,
  ...
)

## S3 method for class 'bounded_reg'
fitted(object, ...)

## S3 method for class 'bounded_reg'
simulate(object, nsim = 1L, seed = NULL, newdata = NULL, trials = NULL, ...)
```

Arguments

x	A bounded_reg fit (for print) or a summary.bounded_reg object (for print.summary).
...	Further arguments; currently unused.
object	A bounded_reg fit.
digits	Number of displayed digits.
statistic	Either the posterior "mean" or "median".
newdata	Optional data frame of predictor values. If NULL, the data used for fitting are used.
type	For predict: "response" returns posterior summaries of the bounded probabilities, "linear" summarizes the linear predictor, and "draws" returns the probability draw matrix.
level	Credible interval level, strictly between 0 and 1.
nsim	Number of posterior predictive simulations.
seed	Optional random seed.
trials	Binomial trial counts for the rows of newdata; defaults to one. For the original data the observed trial counts are used.

Value

- print: the input object, invisibly.
- summary: an object of class summary.bounded_reg containing a table of posterior summaries and MCMC diagnostics.
- coef: a named numeric vector containing both asymptotes and the regression coefficients.
- predict: a data frame with columns estimate, median, lower and upper for type = "response" or "linear"; a matrix of posterior probability draws (draws in rows, observations in columns) for type = "draws".
- fitted: a numeric vector of posterior mean probabilities.
- simulate: a data frame with one column per simulation (sim_1, sim_2, ...) and one row per observation.

Examples

```
set.seed(1)
x <- seq(-3, 3, length.out = 60)
y <- rbounded_binom(-0.5 + 1.2 * x, delta0 = 0.1, delta1 = 0.8)
fit <- bh_fit(y ~ x, data = data.frame(x = x, y = y),
              control = bh_control(iter = 400, warmup = 200, chains = 2),
              seed = 1)
fit
summary(fit)
coef(fit)
predict(fit, newdata = data.frame(x = c(-2, 0, 2)))
head(fitted(fit))
sims <- simulate(fit, nsim = 2, seed = 3)
```

rbounded_binom *Generate bounded binomial outcomes*

Description

Simulate binomial counts whose success probability follows the bounded logistic curve `bounded_probability`.

Usage

```
rbounded_binom(eta, size = 1L, delta0 = 0, delta1 = 1)
```

Arguments

`eta`	Finite numeric linear predictor.
`size`	Nonnegative trial counts, scalar or matching probability length.
`delta0`	Endpoint at eta tending to minus infinity.
`delta1`	Endpoint at eta tending to plus infinity.

Value

A numeric vector of integer-valued counts.

Examples

```
set.seed(1)
rbounded_binom(c(-2, 0, 2), size = 10, delta0 = 0.1, delta1 = 0.8)
```

Index

bh_beta_moments, 2, *8*
bh_control, 3, *5*
bh_diagnose, 3, *5*
bh_fit, *3*, 4, *11*
bh_log_lik, 6
bh_posterior_draws, 7
bh_posterior_predict, 7
bh_prior, *2*, *5*, 8
bounded_probability, 9, *13*
bounded_reg-methods
(print.bounded_reg), 11

coef.bounded_reg (print.bounded_reg), 11

fitted.bounded_reg (print.bounded_reg), 11

model.frame, *5*
model.matrix, *4*, *5*

plot, *10*
plot.bounded_reg, *5*, 10
predict.bounded_reg, *3*
predict.bounded_reg
(print.bounded_reg), 11
print.bh_control (bh_control), 3
print.bh_diagnostic (bh_diagnose), 3
print.bh_prior (bh_prior), 8
print.bounded_reg, 11
print.summary.bounded_reg
(print.bounded_reg), 11

rbounded_binom, 13

simulate.bounded_reg
(print.bounded_reg), 11
summary.bounded_reg
(print.bounded_reg), 11